

BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAAAAAAAAA	CCCCCCCCCCCC	KKK	KKK	UUU	UUU	PPPPPPPPPPPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBBBBBBBBBBBBB		AAA	AAA	CCC	KKKKKKKKKK	UUU	UUU	PPPPPPPPPPPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAAAAAAAAAAAAAAA	CCC	KKK	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBB	BBB	AAA	AAA	CCC	KKK	UUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	UUUUUUUUUUUUUUUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	UUUUUUUUUUUUUUUU	UUU	PPP
BBBBBBBBBBBBBB		AAA	AAA	CCCCCCCCCCCC	KKK	UUUUUUUUUUUUUUUU	UUU	PPP





(2)	93	INIT_NAMEBLOCK, initialize extended name block fields
(3)	167	WILD_DIRECTORY, wildcard output directory processing
(4)	300	INIT_SEL_INFO, initialize selection information
(5)	379	MATCH, match complete file specification
(6)	468	MATCH_FILENAME, match file name, type, version
(7)	559	MATCH_DIRECTORY, match directories
(8)	777	TERMINATE_SCAN, test for scan termination
(9)	856	PARSE_DIRECTORY, parse directory into components

```
0000 1      .TITLE MATCH      File specification matching
0000 2      .IDENT 'V04-000'
0000 3      :---
0000 4      :
0000 5      :*****
0000 6      :
0000 7      :*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8      :*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9      :*  ALL RIGHTS RESERVED.
0000 10     :
0000 11     :*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12     :*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13     :*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14     :*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15     :*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16     :*  TRANSFERRED.
0000 17     :
0000 18     :*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19     :*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20     :*  CORPORATION.
0000 21     :
0000 22     :*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23     :*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24     :
0000 25     :
0000 26     :*****
0000 27     :
0000 28     :++
0000 29     : FACILITY:
0000 30     :     Backup/Restore
0000 31     :
0000 32     : ABSTRACT:
0000 33     :     This module contains file specification matching routines.
0000 34     :
0000 35     : ENVIRONMENT:
0000 36     :     VAX/VMS user mode.
0000 37     :
0000 38     :
0000 39     :--
0000 40     :
0000 41     : AUTHOR: M. Jack, CREATION DATE: 19-Sep-1980
0000 42     : With acknowledgment to Goldstein and Halvorsen for the pieces.
0000 43     :
0000 44     :
0000 45     : MODIFIED BY:
0000 46     :
0000 47     :     V03-002 MLJ0104      Martin L. Jack, 25-Jan-1983 15:02
0000 48     :     Modify MATCH_DIRECTORY to return a bit indicating that the
0000 49     :     terminator contains wildcards, as part of the support for
0000 50     :     execute-only directory scanning, and a bit indicating that the
0000 51     :     terminator is "*", as part of the support for more selective
0000 52     :     directory saving.
0000 53     :
0000 54     :     V03-001 MLJ0085      Martin L. Jack, 30-Mar-1982 13:40
0000 55     :     Modify MATCH to avoid setting a wild directory bit past the
0000 56     :     number of directory levels that actually exist in the
0000 57     :     related file specification.
```



```
0000 58 :
0000 59 :
0000 60 :
0000 61 :
0000 62 :
0000 63 :
0000 64 :
0000 65 :
0000 66 :
0000 67 :
0000 68 :
0000 69 :
0000 70 :
0000 71 :
0000 72 :
0000 73 :
0000 74 :
0000 75 :
0000 76 :
0000 77 :
0000 78 :
0000 79 :
0000 80 :
0000 81 :
0000 82 :
0000 83 :
0000 84 :
0000 85 :
0000 86 :
0000 87 : **
0000 88 :
0000 89 :
0000 90 :
00000000 91
```

V02-007 MLJ0077 Martin L. Jack, 8-Feb-1982 15:11  
Implement negative version numbers.

V02-006 MLJ0063 Martin L. Jack, 11-Dec-1981 14:41  
Support output UIC wildcarding by inserting a comma in the  
related filename string if the selection file specification  
is in UIC format. Modify version number selection so that  
negative pattern versions are accepted and always match.  
Have INIT\_NAMEBLOCK use expanded string if resultant is null.

V02-005 MLJ0054 Martin L. Jack, 6-Nov-1981 16:24  
Correct problem with leading [000000] in candidate directory.  
Modify calling sequence for MATCH\_DIRECTORY to allow long  
filenames.

V02-004 MLJ0036 Martin L. Jack, 28-Aug-1981 16:58  
Extensive rewriting for reel restart and early scan termination.

V02-003 MLJ0025 Martin L. Jack, 8-May-1981 13:42  
Implement latest version selection.

V02-002 MLJ0022 Martin L. Jack, 21-Apr-1981 13:54  
Convert directory wildcarding to use RMS.

V02-001 MLJ0010 Martin L. Jack, 25-Mar-1981 14:04  
Add routines FAST\_MATCH and PARSE\_FAST\_DIR. Correct problems  
with [000000...] and the like as selection filespec.

\$NAMDEF  
.PSECT CODE,EXE,NOWRT



```
0000 93 .SBTTL INIT_NAMEBLOCK, initialize extended name block fields
0000 94
0000 95 :++
0000 96
0000 97 : Functional Description:
0000 98 : This routine initializes the name block descriptors for device,
0000 99 : directory, etc. after BACKUP has constructed a resultant or expanded
0000 100 : string in a name block. The technique used is to scan the string for
0000 101 : terminators since BACKUP constructs the strings in a known format.
0000 102
0000 103 : Calling Sequence:
0000 104 : CALLS/CALLG
0000 105
0000 106 : Input Parameters:
0000 107 : 04(AP) = Pointer to name block.
0000 108
0000 109 : Implicit Inputs:
0000 110 : Resultant string descriptor in name block, if present, otherwise
0000 111 : expanded string descriptor.
0000 112
0000 113 : Output Parameters:
0000 114 : none
0000 115
0000 116 : Implicit Outputs:
0000 117 : Device, directory, etc. descriptors in name block.
0000 118
0000 119 : Routines Called:
0000 120 : none
0000 121
0000 122 : Routine Value:
0000 123 : none
0000 124
0000 125 : Signals:
0000 126 : none
0000 127
0000 128 : Side Effects:
0000 129 : none
0000 130
0000 131 :--
0000 132
001C 0000 133 .ENTRY INIT_NAMEBLOCK,*M<R2,R3,R4>
54 04 AC D0 0002 134 MOVL 4(AP),R4 : Get pointer to name block
53 04 A4 D0 0006 135 MOVL NAMSL_RSA(R4),R3 : Get resultant string descriptor
52 03 A4 9A 000A 136 MOVZBL NAMSB_RSL(R4),R2
53 0C A4 D0 0010 137 BNEQ 5$ : Branch if resultant present
52 0B A4 9A 0014 138 MOVL NAMSL_ESA(R4),R3 : Get expanded string descriptor
40 A4 53 D0 0018 139 MOVZBL NAMSB_ESL(R4),R2
63 52 5B 8F 3A 001F 140 5$: MOVL R3,NAMSL_NODE(R4) : Set node address
53 0C A4 D0 001C 141 CLRB NAMSB_NODE(R4) : Set node length
39 A4 52 50 83 002E 142 LOCC #'A'[*,R2,(R3) : Scan for device terminator
63 52 5D 8F 3A 0036 143 BNEQ 10$ : Branch if found
54 04 AC D0 0024 144 LOCC #'A'<*,R2,(R3) : Scan for alternate terminator
44 A4 53 D0 002A 145 10$: MOVL R3,NAMSL_DEV(R4) : Set device address
39 A4 52 50 83 002E 146 SUBB3 R0,R2,NAMSB_DEV(R4) : Set device length
63 52 5D 8F 3A 0033 147 MOVQ R0,R2 : Prune device from string
54 04 AC D0 0036 148 LOCC #'A']',R2,(R3) : Scan for directory terminator
53 0C A4 D0 003B 149 BNEQ 20$ : Branch if found
```



63	52	3E	3A	003D	150	LOCC	#^A'>',R2,(R3)	; Scan for alternate terminator
		50	D7	0041	151	DECL	R0	; Adjust to include terminator
		51	D6	0043	152	INCL	R1	
48	A4	53	D0	0045	153	MOVL	R3,NAM\$L_DIR(R4)	; Set directory address
3A	A4	52	83	0049	154	SUBB3	R0,R2,NAM\$B_DIR(R4)	; Set directory length
		50	7D	004E	155	MOVQ	R0,R2	; Prune directory from string
63	52	2E	3A	0051	156	LOCC	#^A'.' ,R2,(R3)	; Scan for file name terminator
4C	A4	53	D0	0055	157	MOVL	R3,NAM\$L_NAME(R4)	; Set file name address
3B	A4	52	83	0059	158	SUBB3	R0,R2,NAM\$B_NAME(R4)	; Set file name length
		50	7D	005E	159	MOVQ	R0,R2	; Prune file name from string
63	52	3B	3A	0061	160	LOCC	#^A':',R2,(R3)	; Scan for file type terminator
50	A4	53	D0	0065	161	MOVL	R3,NAM\$L_TYPE(R4)	; Set file type address
3C	A4	52	83	0069	162	SUBB3	R0,R2,NAM\$B_TYPE(R4)	; Set file type length
54	A4	51	D0	006E	163	MOVL	R1,NAM\$L_VER(R4)	; Set file version address
3D	A4	50	90	0072	164	MOVQ	R0,NAM\$B_VER(R4)	; Set file version length
			04	0076	165	RET		; Return

```
0077 167 .SBTTL WILD_DIRECTORY, wildcard output directory processing
0077 168
0077 169 :++
0077 170 :
0077 171 : Functional Description:
0077 172 : This routine sets up the necessary status bits in the name block for
0077 173 : RMS wildcard output directory processing. If necessary, it also adds
0077 174 : a comma to the resultant filename string.
0077 175 :
0077 176 : Calling Sequence:
0077 177 : CALLS/CALLG
0077 178 :
0077 179 : Input Parameters:
0077 180 : 04(AP) = Address of descriptor for selection file specification.
0077 181 : 08(AP) = Address of name block.
0077 182 :
0077 183 : Implicit Inputs:
0077 184 : none
0077 185 :
0077 186 : Output Parameters:
0077 187 : none
0077 188 :
0077 189 : Implicit Outputs:
0077 190 : NAMS_L_FNB contains the wildcard directory bit corresponding to the
0077 191 : leftmost wild directory in the selection file specification, if any.
0077 192 : If the selection file specification is in UIC format, a comma is
0077 193 : inserted in the resultant filename string.
0077 194 :
0077 195 : Routines Called:
0077 196 : PARSE_DIRECTORY
0077 197 :
0077 198 : Routine Value:
0077 199 : none
0077 200 :
0077 201 : Signals:
0077 202 : none
0077 203 :
0077 204 : Side Effects:
0077 205 : none
0077 206 :
0077 207 :--
0077 208
00FC 0077 209 .ENTRY WILD_DIRECTORY, ^M<R2,R3,R4,R5,R6,R7>
0079 210 :
0079 211 : Parse the resultant file specification to obtain the number of directories.
0079 212 :
0079 213 : MOVAB -132(SP), SP ; Make result space on stack
007E 214 : MOVL 8(AP), R6 ; Point to name block
0082 215 : MOVZBL NAMS_B_RSL(R6), R2 ; Get resultant string descriptor
0086 216 : MOVL NAMS_L_RSA(R6), R3
008A 217 : LOCC #^A'[', R2, (R3) ; Scan for start of directory
008F 218 : BNEQ 5$, ; Branch if found
0091 219 : LOCC #^A'<', R2, (R3) ; Scan for alternate syntax
0095 220 5$: MOVAB -(R0), R2 ; Prune beginning of string
0098 221 : MOVAB 1(R1), R3
009C 222 : LOCC #^A']', R2, (R3) ; Scan for end of directory
00A1 223 : BNEQ 7$, ; Branch if found
```



```
63 52 3E 3A 00A3 224 LOCC #^A'>',R2,(R3) ; Scan for alternate syntax
52 50 C2 00A7 225 7$: SUBL2 R0,R2 ; Prune end of string
54 5E D0 00AA 226 MOVL SP,R4 ; Point to it
57 0373 30 00AD 227 BSBW PARSE_DIRECTORY ; Parse the directory specification
6E 3C 00B0 228 MOVZWL (SP),R7 ; Get number of levels
00B3 229 ;
00B3 230 ; Parse the selection file specification.
00B3 231 ;
63 52 04 BC 7D 00B3 232 MOVQ @4(AP),R2 ; Get selection string descriptor
52 5B 8F 3A 00B7 233 LOCC #^A'[',R2,(R3) ; Scan for start of directory
04 12 00BC 234 BNEQ 10$ ; Branch if found
63 52 3C 3A 00BE 235 LOCC #^A'<',R2,(R3) ; Scan for alternate syntax
52 70 9E 00C2 236 10$: MOVAB -(R0),R2 ; Prune beginning of string
53 01 A1 9E 00C5 237 MOVAB 1(R1),R3 ;
63 52 5D 8F 3A 00C9 238 LOCC #^A']',R2,(R3) ; Scan for end of directory
04 12 00CE 239 BNEQ 20$ ; Branch if found
63 52 3E 3A 00D0 240 LOCC #^A'>',R2,(R3) ; Scan for alternate syntax
52 50 C2 00D4 241 20$: SUBL2 R0,R2 ; Prune end of string
54 5E D0 00D7 242 MOVL SP,R4 ; Point to it
0346 30 00DA 243 BSBW PARSE_DIRECTORY ; Parse the directory specification
00DD 244 ;
00DD 245 ; Initialize for loop.
00DD 246 ;
56 08 AC 34 C1 00DD 247 ADDL3 #NAMS_L_FNB,8(AP),R6 ; Point to NAMS_L_FNB
66 D4 00E2 248 CLRL (R6) ; Clear FNB
29 6E 10 E0 00E4 249 BBS #16,(SP),60$ ; Branch if UIC-format directory
54 8E D0 00E8 250 MOVL (SP)+,R4 ; Get count of directories
55 18 D0 00EB 251 MOVL #NAMS_V_WILD_UFD,R5 ; Get first bit number
00EE 252 ;
00EE 253 ; Loop over all directories searching for the leftmost one that
00EE 254 ; contains a wildcard.
00EE 255 ;
57 D7 00EE 256 30$: DECL R7 ; Count resultant directory levels
1E 19 00F0 257 BLSS 50$ ; Br if none left to avoid setting bit
52 8E 7D 00F2 258 MOVQ (SP)+,R2 ; Get next directory descriptor
2E 63 91 00F5 259 CMPB (R3),#^A'.' ; Was it ellipsis?
12 13 00F8 260 BEQL 40$ ; Branch if yes
63 52 2A 3A 00FA 261 LOCC #^A'*',R2,(R3) ; Contains asterisk?
0C 12 00FE 262 BNEQ 40$ ; Branch if yes
63 52 25 3A 0100 263 LOCC #^A'%',R2,(R3) ; Contains percent?
06 12 0104 264 BNEQ 40$ ; Branch if yes
55 D6 0106 265 INCL R5 ; Increment bit number
E3 54 F5 0108 266 SOBGTR R4,30$ ; Loop for all directories
04 010B 267 RET ; Return
010C 268 ;
010C 269 ; Here if a wildcard level is found to set the corresponding bit.
010C 270 ;
00 66 55 E2 010C 271 40$: BBSS R5,(R6),50$ ; Set wildcard bit
04 0110 272 50$: RET ; Return
0111 273 ;
0111 274 ; Here if a UIC-format directory. PARSE_DIRECTORY has transformed a '*' in
0111 275 ; the group or member portion into '%%', so here we just test for a
0111 276 ; leading '%' in the appropriate position.
0111 277 ;
00 66 13 E2 0111 278 60$: BBSS #NAMS_V_GRP_MBR,(R6),70$ ; Set UIC-format bit
50 08 AE D0 0115 279 70$: MOVL 8(SP),R0 ; Point to directory string
25 60 91 0119 280 CMPB (R0),#^A'%' ; Group part wild?
```

```
00 66 04 12 011C 281      BNEQ 80$      ; Branch if no
25 03 18 E2 011E 282      BBSS #NAMS_V_WILD_GRP,(R6),80$; Set wild group bit
04 12 0122 283 80$:      CMPB 3(R0),#^A'^%      ; Member part wild?
00 66 04 12 0126 284      BNEQ 90$      ; Branch if no
19 E2 0128 285      BBSS #NAMS_V_WILD_MBR,(R6),90$; Set wild member bit
      012C 286      ;
      012C 287      ; Insert a comma between the group and member numbers in the related filename
      012C 288      ; string to compensate for the fact that directory specifications are stored in
      012C 289      ; the save set without a comma. RMS requires the comma for proper operation of
      012C 290      ; output UIC wildcarding.
      012C 291      ;
      012C 292 90$:      INCB NAMS_B_RSL-NAMS_L_FNB(R6) ; Increase length of related filename
50  CF A6 96 012C 293      MOVZBL NAMS_B_RSL-NAMS_L_FNB(R6),R0 ; Get length
      CF A6 9A 012F 294      SUBL2 #5,R0      ; Allow for 'Lnnn' and INCB
56 50 05 C2 0133 295      MOVL NAMS_L_RSA-NAMS_L_FNB(R6),R6 ; Point to related filename string
05 A6 04 A6 50 28 013A 296      MOVC3 R0,4(R6),5(R6) ; Slide related filename string over
04 A6 2C 90 0140 297      MOVB #^A',',4(R6) ; Insert a comma
      04 0144 298      RET
```



```
0145 300      .SBTTL INIT_SEL_INFO, initialize selection information
0145 301
0145 302      ++
0145 303
0145 304      Functional Description:
0145 305          This routine generates necessary information concerning a selection
0145 306          file specification.
0145 307
0145 308      Calling Sequence:
0145 309          CALLS/CALLG
0145 310
0145 311      Input Parameters:
0145 312          04(AP) = Address of descriptor for selection file specification.
0145 313
0145 314      Implicit Inputs:
0145 315          none
0145 316
0145 317      Output Parameters:
0145 318          08(AP) = Address of a quadword that receives a descriptor for the
0145 319          directory portion of the file specification.
0145 320          12(AP) = Address of a quadword that receives a descriptor for the
0145 321          name, type, and version portion of the file specification.
0145 322          16(AP) = Address of a longword that receives the zero or negative
0145 323          version number if the file specification selects a latest
0145 324          version and a positive number otherwise.
0145 325
0145 326      Implicit Outputs:
0145 327          none
0145 328
0145 329      Routines Called:
0145 330          LIB$CVT_DTB
0145 331
0145 332      Routine Value:
0145 333          none
0145 334
0145 335      Signals:
0145 336          none
0145 337
0145 338      Side Effects:
0145 339          none
0145 340
0145 341      --
0145 342
000C 0145 343      .ENTRY INIT_SEL_INFO, ^M<R2,R3>
0147 344
0147 345      Parse the string into a directory string and a
0147 346      name, type, and version string.
0147 347
0147 348      MOVQ    @4(AP),R2          : Get string descriptor
0147 349      LOCC    #^A'[',R2,(R3)    : Scan for start of directory
0147 350      BNEQ    10$,R2,(R3)       : Branch if found
0147 351      LOCC    #^A'<',R2,(R3)    : Scan for alternate syntax
0147 352      MOVAB   -(R0),R2          : Prune beginning of string
0147 353      MOVAB   1(R1),R3
0147 354      LOCC    #^A']',R2,(R3)    : Scan for end of directory
0147 355      BNEQ    20$,R2,(R3)       : Branch if found
0147 356      LOCC    #^A'>',R2,(R3)    : Scan for alternate syntax
```

63	52	04	BC	7D	0147	348	MOVQ	@4(AP),R2	:	Get string descriptor
	52	5B	8F	3A	014B	349	LOCC	#^A'[,R2,(R3)	:	Scan for start of directory
			04	12	0150	350	BNEQ	10\$,R2,(R3)	:	Branch if found
63	52	52	3C	3A	0152	351	LOCC	#^A'<',R2,(R3)	:	Scan for alternate syntax
			70	9E	0156	352	MOVAB	-(R0),R2	:	Prune beginning of string
	53	01	A1	9E	0159	353	MOVAB	1(R1),R3	:	
63	52	5D	8F	3A	015D	354	LOCC	#^A']',R2,(R3)	:	Scan for end of directory
			04	12	0162	355	BNEQ	20\$,R2,(R3)	:	Branch if found
63	52	3E	3A	0164	356	LOCC	#^A'>',R2,(R3)	:	Scan for alternate syntax	

```
0168 357 ;  
0168 358 : Construct and return the two descriptors.  
0168 359 :  
08 52 50 C2 0168 360 20$:  SUBL2  R0,R2      : R2-R3: directory descriptor  
BC 52 7D 0168 361      MOVQ   R2,@8(AP)    : Set directory descriptor  
50 D7 0168 362      DECL   R0             : R0-R1: name, type, version descriptor  
51 D6 0171 363      INCL   R1             :  
0C BC 50 7D 0173 364      MOVQ   R0,@12(AP)  : Set name, type, version descriptor  
0177 365 :  
0177 366 : Compute the latest-version flag.  
0177 367 :  
10 BC 01 D0 0177 368      MOVL   #1,@16(AP)  : Assume not latest version  
61 50 3B 3A 017B 369      LOCC   #'A';',R0,(R1) : Scan for ','  
15 13 017F 370      BEQL   30$           : Branch if not found  
7E DF 0181 371      PUSHAL -(SP)         : Push address of result location  
01 A1 9F 0183 372      PUSHAB 1(R1)       : Push descriptor for version number  
70 9F 0186 373      PUSHAB -(R0)         :  
00000000'GF 03 FB 0188 374      CALLS #3,G^LIB$CVT_DTB : Convert pattern version number  
04 50 E9 018F 375      BLBC   R0,30$      : Branch if conversion failed  
10 BC 8E D0 0192 376      MOVL   (SP)+,@16(AP) : Get pattern version number  
04 0196 377 30$:  RET                    : Return
```



```
0197 379 .SBTTL MATCH, match complete file specification
0197 380
0197 381 :++
0197 382 :
0197 383 : Functional Description:
0197 384 : This routine executes a wild card match on a candidate and pattern
0197 385 : file specification.
0197 386 :
0197 387 : Calling Sequence:
0197 388 : CALLS/CALLG
0197 389 :
0197 390 : Input Parameters:
0197 391 : 04(AP) = Address of descriptor for candidate file specification.
0197 392 : 08(AP) = Address of descriptor for pattern file specification.
0197 393 :
0197 394 : Implicit Inputs:
0197 395 : none
0197 396 :
0197 397 : Output Parameters:
0197 398 : none
0197 399 :
0197 400 : Implicit Outputs:
0197 401 : none
0197 402 :
0197 403 : Routines Called:
0197 404 : MATCH_DIRECTORY
0197 405 : MATCH_FILENAME
0197 406 :
0197 407 : Routine Value:
0197 408 : True if the strings match.
0197 409 :
0197 410 : Signals:
0197 411 : none
0197 412 :
0197 413 : Side Effects:
0197 414 : none
0197 415 :
0197 416 :--
000C 0197 417
0197 418 .ENTRY MATCH, ^M<R2,R3>
0199 419 :
0199 420 : Parse the candidate string into a directory string and a
0199 421 : name, type, and version string.
0199 422 :
0199 423 :
63 52 04 BC 7D 0199 423 MOVQ @4(AP),R2 : Get candidate string descriptor
52 52 5B 8F 3A 019D 424 LOCC #^A'[',R2,(R3) : Scan for start of directory
04 12 01A2 425 BNEQ 10$ : Branch if found
63 52 3C 3A 01A4 426 LOCC #^A'<',R2,(R3) : Scan for alternate syntax
52 52 70 9E 01A8 427 10$: MOVAB -(R0),R2 : Prune beginning of string
53 01 A1 9E 01AB 428 MOVAB 1(R1),R3
63 52 5D 8F 3A 01AF 429 LOCC #^A']',R2,(R3) : Scan for end of directory
04 12 01B4 430 BNEQ 20$ : Branch if found
63 52 3E 3A 01B6 431 LOCC #^A'>',R2,(R3) : Scan for alternate syntax
53 DD 01BA 432 20$: PUSHL R3 : Make directory descriptor
7E 52 50 C3 01BC 433 SUBL3 R0,R2,-(SP)
01 A1 9F 01C0 434 PUSHAB 1(R1) : Make name, type, version descriptor
70 9F 01C3 435 PUSHAB -(R0)
```

```
01C5 436 :  
01C5 437 : Parse the pattern string into a directory string and a  
01C5 438 : name, type, and version string.  
01C5 439 :  
63 52 08 BC 7D 01C5 440 : MOVQ @8(AP),R2 : Get pattern string descriptor  
52 5B BF 3A 01C9 441 : LOCC #^A'[',R2,(R3) : Scan for start of directory  
04 12 01CE 442 : BNEQ 30$ : Branch if found  
63 52 3C 3A 01D0 443 : LOCC #^A'<',R2,(R3) : Scan for alternate syntax  
52 70 9E 01D4 444 30$: MOVAB -(R0),R2 : Prune beginning of string  
53 01 A1 9E 01D7 445 : MOVAB 1(R1),R3 :  
63 52 5D BF 3A 01DB 446 : LOCC #^A']',R2,(R3) : Scan for end of directory  
04 12 01E0 447 : BNEQ 40$ : Branch if found  
63 52 3E 3A 01E2 448 : LOCC #^A'>',R2,(R3) : Scan for alternate syntax  
53 DD 01E6 449 40$: PUSHL R3 : Make directory descriptor  
7E 52 50 C3 01E8 450 : SUBL3 R0,R2,-(SP) :  
01 A1 9F 01EC 451 : PUSHAB 1(R1) : Make name, type, version descriptor  
70 9F 01EF 452 : PUSHAB -(R0) :  
01F1 453 :  
01F1 454 : Match the name, type, and version strings.  
01F1 455 :  
5E DD 01F1 456 : PUSHL SP : Push pattern descriptor  
14 AE 9F 01F3 457 : PUSHAB 20(SP) : Push candidate descriptor  
09'AF 02 FB 01F6 458 : CALLS #2,B^MATCH_FILENAME : Match filenames  
0B 50 E9 01FA 459 : BLBC R0,50$ : Branch if failed  
01FD 460 :  
01FD 461 : Match the directory strings.  
01FD 462 :  
08 AE 9F 01FD 463 : PUSHAB 8(SP) : Push pattern descriptor  
1C AE 9F 0200 464 : PUSHAB 28(SP) : Push candidate descriptor  
027A'CF 02 FB 0203 465 : CALLS #2,W^MATCH_DIRECTORY : Match directories  
04 0208 466 50$: RET : Return
```



```
0209 468 .SBTTL MATCH_FILENAME, match file name, type, version
0209 469
0209 470 :++
0209 471 :
0209 472 : Functional Description:
0209 473 : This routine executes a wild card match on the name, type, and version
0209 474 : of a candidate and pattern file specification. If the pattern selects
0209 475 : latest version, this routine assumes that the candidate file is the
0209 476 : latest version, and therefore that the version matches.
0209 477 :
0209 478 : Calling Sequence:
0209 479 : CALLS/CALLG
0209 480 :
0209 481 : Input Parameters:
0209 482 : 04(AP) = Address of descriptor for candidate file name, type, version.
0209 483 : 08(AP) = Address of descriptor for pattern file name, type, version.
0209 484 :
0209 485 : Implicit Inputs:
0209 486 : none
0209 487 :
0209 488 : Output Parameters:
0209 489 : none
0209 490 :
0209 491 : Implicit Outputs:
0209 492 : none
0209 493 :
0209 494 : Routines Called:
0209 495 : FMGSMATCH_NAME
0209 496 : LIB$CVT_DTB
0209 497 :
0209 498 : Routine Value:
0209 499 : True if the strings match.
0209 500 :
0209 501 : Signals:
0209 502 : none
0209 503 :
0209 504 : Side Effects:
0209 505 : none
0209 506 :
0209 507 :--
0209 508
03Fc 0209 509 .ENTRY MATCH_FILENAME, ^M<R2,R3,R4,R5,R6,R7,R8,R9>
0208 510 :
0208 511 : Parse the candidate string into a name and type string and a version string.
0208 512 :
0208 513 : MOVQ 04(AP),R2 ; Get candidate string descriptor
020F 514 : MOVZWL R2,R2 ; Truncate length to a word
0212 515 : LOCC #'A';',R2,(R3) ; Scan for version number
0216 516 : BEQL 20$ ; Branch if not found
0218 517 : SUBL2 R0,R2 ; R2-R3: candidate name and type
0218 518 : MOVAB 1(R1),R7 ; R6-R7: candidate version
021F 519 : MOVAB -(R0),R6 ;
0222 520 :
0222 521 : Parse the pattern string into a name and type string and a version string.
0222 522 :
0222 523 : MOVQ 08(AP),R4 ; Get pattern string descriptor
0226 524 : MOVZWL R4,R4 ; Truncate length to a word
```

52 04 BC 7D  
63 52 52 3C  
52 3B 3A  
61 13  
52 50 C2  
57 01 A1 9E  
56 70 9E

54 08 BC 7D  
54 54 3C

```
65 54 3B 3A 0229 525 LOCC #^A';',R4,(R5) ; Scan for version number
    54 4A 13 022D 526 BEQL 20$ ; Branch if not found
    59 54 50 C2 022F 527 SUBL2 R0,R4 ; R4-R5: pattern name and type
    01 A1 9E 0232 528 MOVAB 1(R1),R9 ; R8-R9: pattern version
    58 70 9E 0236 529 MOVAB -(R0),R8 ;
    0239 530 ;
    0239 531 ; Match the file name and type.
    0239 532 ;
    00000000'GF 16 0239 533 JSB G^FMG$MATCH_NAME ; Match name and type
    37 50 E9 023F 534 BLBC R0,20$ ; Branch if failed
    0242 535 ;
    0242 536 ; The match succeeded. Now the version numbers must be matched.
    0242 537 ;
    58 D5 0242 538 TSTL R8 ; Pattern version null?
    30 13 0244 539 BEQL 10$ ; Branch if yes
    2A 69 91 0246 540 CMPB (R9),#^A'*' ; Wildcard pattern version?
    2B 13 0249 541 BEQL 10$ ; Branch if yes
    7E DF 024B 542 PUSHAL -(SP) ; Push address of result location
    00000000'GF 58 7D 024D 543 MOVQ R8,-(SP) ; Push descriptor
    1F 03 FB 0250 544 CALLS #3,G^LIB$CVT_DTB ; Convert pattern version number
    58 50 E9 0257 545 BLBC R0,20$ ; Branch if conversion failed
    17 15 025D 546 MOVL (SP)+,R8 ; Get pattern version number
    7E DF 025F 547 BLEQ 10$ ; Branch if explicit zero or negative
    7E 56 7D 0261 548 PUSHAL -(SP) ; Push address of result location
    00000000'GF 03 FB 0264 549 MOVQ R6,-(SP) ; Push descriptor
    0B 50 E9 026B 550 CALLS #3,G^LIB$CVT_DTB ; Convert candidate version number
    50 50 D4 026E 551 BLBC R0,20$ ; Branch if conversion failed
    58 8E D1 0270 552 CLRL R0 ; Assume failure return
    01 13 0273 553 CMPL (SP)+,R8 ; Compare versions
    04 0275 554 BEQL 10$ ; Branch if versions equal
    50 01 D0 0276 555 RET ; Return failure
    04 0279 556 10$: MOVL #1,R0 ; Set success status
    20$: RET ; Return
```



```
027A 559 .SBTTL MATCH_DIRECTORY, match directories
027A 560
027A 561 :++
027A 562
027A 563 : Functional Description:
027A 564 : This routine executes a wild card match on the directory
027A 565 : of a candidate and pattern file specification.
027A 566
027A 567 : Calling Sequence:
027A 568 : CALLS/CALLG
027A 569
027A 570 : Input Parameters:
027A 571 : 04(AP) = Address of descriptor for candidate directory.
027A 572 : 08(AP) = Address of descriptor for pattern directory.
027A 573 : 20(AP) = Optional address of descriptor for pattern
027A 574 : name, type, and version.
027A 575
027A 576 : Implicit Inputs:
027A 577 : none
027A 578
027A 579 : Output Parameters:
027A 580 : 12(AP) = Optional address of descriptor to receive terminator file
027A 581 : name and type.
027A 582 : 16(AP) = Optional address of word to receive terminator file version
027A 583 : number.
027A 584
027A 585 : Implicit Outputs:
027A 586 : none
027A 587
027A 588 : Routines Called:
027A 589 : FMG$MATCH_NAME
027A 590 : LIB$CVT_DTB
027A 591 : PARSE_DIRECTORY
027A 592
027A 593 : Routine Value:
027A 594 : Bit 0: Set if the directory strings match.
027A 595 : Bit 1: Set if the directory must be scanned.
027A 596 : Bit 2: Set if the terminator contains wildcards.
027A 597 : Bit 3: Set if the terminator is "*" or equivalent.
027A 598
027A 599 : Signals:
027A 600 : none
027A 601
027A 602 : Side Effects:
027A 603 : none
027A 604
027A 605 :--
027A 606
027A 607 .ENTRY MATCH_DIRECTORY, ^M<R2,R3,R4,R5,R6,R7,R8,R9>
SE FF38 CE 03FC 027C 608 MOVAB -200(SP),SP ; Make directory parse space on stack
0281 609
0281 610 : Parse the candidate directory string.
0281 611
0281 612 : MOVQ @4(AP),R2 ; Get candidate string descriptor
52 04 BC 7D 0285 613 MOVZWL R2,R2 ; Truncate length to a word
52 52 52 3C 0288 614 MOVL SP,R4 ; Point to result area
54 5E D0 028B 615 BSBW PARSE_DIRECTORY ; Parse the specification
0195 30
```



```

028E 616 :
028E 617 : Parse the pattern directory string.
028E 618 :
52 08 BC 7D 028E 619 : MOVQ @8(AP),R2 ; Get pattern string descriptor
52 52 52 3C 0292 620 : MOVZWL R2,R2 ; Truncate length to a word
54 44 AE 9E 0295 621 : MOVAB 68(SP),R4 ; Point to result area
0187 30 0299 622 : BSBW PARSE_DIRECTORY ; Parse the specification
029C 623 :
029C 624 : Set up for matching directories.
029C 625 :
53 52 6E 3C 029C 626 : MOVZWL (SP),R2 ; Get candidate descriptor count
53 04 AE 9E 029F 627 : MOVAB 4(SP),R3 ; Point to candidate string results
54 44 AE 3C 02A3 628 : MOVZWL 68(SP),R4 ; Get pattern descriptor count
55 48 AE 9E 02A7 629 : MOVAB 72(SP),R5 ; Point to first pattern descriptor
56 7C 02AB 630 : CLRQ R6 ; Clear saved directory count, pointer
02AD 631 :
02AD 632 : A special matching rule applies if the pattern string directory is in
02AD 633 : UIC format. Such a directory matches only if the target string
02AD 634 : directory contains six octal digits.
02AD 635 :
19 46 AE E9 02AD 636 : BLBC 70(SP),20$ ; Branch if pattern not UIC format
01 52 D1 02B1 637 : CMPL R2,#1 ; Exactly one directory in target?
45 12 02B4 638 : BNEQ 50$ ; Branch if no
50 63 7D 02B6 639 : MOVQ (R3),R0 ; Get target directory descriptor
06 50 D1 02B9 640 : CMPL R0,#6 ; Six characters?
3D 12 02BC 641 : BNEQ 50$ ; Branch if no
59 81 30 83 02BE 642 10$: SUBB3 #^A'0',(R1)+,R9 ; Bias character by ASCII 0
07 59 91 02C2 643 : CMPB R9,#7 ; Make sure in range
34 1A 02C5 644 : BGTRU 50$ ; Branch if no
F4 50 F5 02C7 645 : SOBGTR R0,10$ ; Loop until all examined
02CA 646 :
02CA 647 : Handle MFD in pattern and candidate strings.
02CA 648 :
06 65 D1 02CA 649 20$: CMPL (R5),#6 ; First pattern directory length 6?
1A 12 02CD 650 : BNEQ 40$ ; Branch if no
04 B5 06 30 3B 02CF 651 : SKPC #^A'0',#6,@4(R5) ; First pattern directory '000000'?
13 12 02D4 652 : BNEQ 40$ ; Branch if no
06 63 D1 02D6 653 : CMPL (R3),#6 ; First candidate directory length 6?
07 12 02D9 654 : BNEQ 30$ ; Branch if no
04 B3 06 30 3B 02DB 655 : SKPC #^A'0',#6,@4(R3) ; First candidate directory '000000'?
1C 13 02E0 656 : BEQL 60$ ; Branch if yes
54 D7 02E2 657 30$: DECL R4 ; Prune '000000' from pattern
55 08 C0 02E4 658 : ADDL2 #8,R5
15 11 02E7 659 : BRB 60$ ; Go to do full match
06 63 D1 02E9 660 40$: CMPL (R3),#6 ; First candidate directory length 6?
10 12 02EC 661 : BNEQ 60$ ; Branch if no
04 B3 06 30 3B 02EE 662 : SKPC #^A'0',#6,@4(R3) ; Candidate directory is '000000'?
09 12 02F3 663 : BNEQ 60$ ; Branch if no
50 65 7D 02F5 664 : MOVQ (R5),R0 ; Get descriptor for first pattern
00AC 31 02F8 665 45$: BRW 140$ ; Branch to fail with scan
00A6 31 02FB 666 50$: BRW 130$ ; Branch to fail
02FE 667 :
02FE 668 : Now execute the full-scale match.
02FE 669 :
54 D7 02FE 670 60$: DECL R4 ; Pattern exhausted?
34 19 0300 671 : BLSS 80$ ; Branch if yes
50 85 7D 0302 672 : MOVQ (R5)+,R0 ; Get next directory in pattern
```



```
2E 61 91 0305 673 CMPB (R1),#^A'. ' ; Check for ellipsis
    03 12 0308 674 BNEQ 61$ ; Branch
0086 31 030A 675 BRW 110$ ; if yes
    52 D7 030D 676 61$: DECL R2 ; Candidate exhausted?
    E7 19 030F 677 BLSS 45$ ; Branch if yes
    3C BB 0311 678 PUSHR #^M<R2,R3,R4,R5> ; Save registers around MATCH_NAME
    54 50 7D 0313 679 MOVQ R0,R4 ; Load pattern descriptor to R4-R5
    52 63 7D 0316 680 MOVQ (R3),R2 ; Load candidate descriptor to R2-R3
00000000'GF 16 0319 681 JSB G^FMG$MATCH_NAME ; Check candidate against pattern
    3C BA 031F 682 POPR #^M<R2,R3,R4,R5> ; Restore registers
    53 08 C0 0321 683 ADDL2 #8,R3 ; Advance past descriptor
    D7 50 E8 0324 684 BLBS R0,60$ ; Branch if match
    0327 685 ;
    0327 686 ; We have detected a mismatch, or we are out of pattern string while there
    0327 687 ; is input left. Back up to the last ellipsis, advance a directory of the
    0327 688 ; input, and try again.
    0327 689 ;
    56 D7 0327 690 70$: DECL R6 ; Count a directory from saved input
    75 19 0329 691 BLSS 120$ ; Branch if no saved input
    57 08 C0 032B 692 ADDL2 #8,R7 ; Set to try next input directory
    52 56 7D 032E 693 MOVQ R6,R2 ; Restore pointers to backup point
    54 58 7D 0331 694 MOVQ R8,R4 ; to retry matching
    C8 11 0334 695 BRB 60$ ; Continue testing
    0336 696 ;
    0336 697 ; Here when pattern string is exhausted.
    0336 698 ;
    52 D5 0336 699 80$: TSTL R2 ; Input exhausted?
    ED 12 0338 700 BNEQ 70$ ; Branch if no
    033A 701 ;
    033A 702 ; Establish the pattern file name, type, and version specified by
    033A 703 ; the fourth parameter as the terminator and return success.
    033A 704 ;
    58 0F D0 033A 705 90$: MOVL #15,R8 ; Establish return value
    05 6C 91 033D 706 CMPB (AP),#5 ; Parameters present?
    4D 1F 0340 707 BLSSU 100$ ; Branch if no
    59 0C AC D0 0342 708 MOVL 12(AP),R9 ; Point to result area descriptor
    69 B4 0346 709 CLRW (R9) ; Set no terminator
    57 D5 0348 710 TSTL R7 ; Ellipsis found in string?
    43 12 034A 711 BNEQ 100$ ; Branch if yes
    52 14 BC 7D 034C 712 MOVQ @20(AP),R2 ; Get descriptor for specification
    52 52 3C 0350 713 MOVZWL R2,R2 ; Truncate length to a word
    63 52 2A 3A 0353 714 LOCC #^A'*,R2,(R3) ; Asterisk wildcard in string?
    09 12 0357 715 BNEQ 95$ ; Branch if found
    63 52 25 3A 0359 716 LOCC #^A'%',R2,(R3) ; Percent wildcard in string?
    03 12 035D 717 BNEQ 95$ ; Branch if found
    58 04 CA 035F 718 BICL2 #4,R8 ; Indicate nonwild terminator string
    63 2A 91 0362 719 95$: CMPB #^A'*,(R3) ; Leading asterisk wildcard?
    03 13 0365 720 BEQL 96$ ; Branch if yes
    58 08 CA 0367 721 BICL2 #8,R8 ; Indicate non-asterisk terminator
    63 52 3B 3A 036A 722 96$: LOCC #^A':',R2,(R3) ; Scan for version number
    1F 13 036E 723 BEQL 100$ ; Branch if not found
    52 50 C2 0370 724 SUBL2 R0,R2 ; R2-R3: name and type
    00 DD 0373 725 PUSHL #0 ; Create and clear result location
    5E DD 0375 726 PUSHL SP ; Push address of result location
    01 A1 9F 0377 727 PUSHAB 1(R1) ; Push address of version number
    70 9F 037A 728 PUSHAB -(R0) ; Push length of version number
00000000'GF 03 FB 037C 729 CALLS #3,G^LIB$CVT_DTB ; Convert version number
```



```
10 BC 8E F7 0383 730 CVTLW (SP)+,a16(AP) ; Set terminator file version
04 B9 69 52 B0 0387 731 MOVW R2,(R9) ; Set byte count of name and type
63 63 52 28 038A 732 MOVW R2,(R3),a4(R9) ; Set terminator file name and type
50 58 D0 038F 733 100$: MOVL R8,R0 ; Set status
04 0392 734 RET ; Return
0393 735 ;
0393 736 ; We have encountered an ellipsis in the pattern string. Save the string
0393 737 ; pointers for backup and retry.
0393 738 ;
56 52 7D 0393 739 110$: MOVQ R2,R6 ; Save current string pointers
58 54 7D 0396 740 MOVQ R4,R8 ; of both strings
54 D5 0399 741 TSTL R4 ; Pattern string null after ellipsis?
9D 13 C39B 742 BEQL 90$ ; Ellipsis at end matches all
FF5E 31 039D 743 BRW 60$ ; Continue testing
03A0 744 ;
03A0 745 ; Here when match fails.
03A0 746 ;
57 D5 03A0 747 120$: TSTL R7 ; Ellipsis found in string?
03 12 03A2 748 BNEQ 140$ ; Branch if yes
50 D4 03A4 749 130$: CLRL R0 ; Set failure
04 03A6 750 RET ;
03A7 751 ;
03A7 752 ; Here to establish the directory specified by R0-R1 as the terminator.
03A7 753 ;
58 0E D0 03A7 754 140$: MOVL #14,R8 ; Establish return status
04 6C 91 03AA 755 CMPB (AP),#4 ; Parameters present?
38 1F 03AD 756 BLSSU 150$ ; Branch if no
59 0C AC D0 03AF 757 MOVL 12(AP),R9 ; Point to result area descriptor
69 B4 03B3 758 CLRW (R9) ; Set no terminator
57 D5 03B5 759 TSTL R7 ; Ellipsis found in string?
2E 12 03B7 760 BNEQ 150$ ; Branch if yes
52 50 7D 03B9 761 MOVQ R0,R2 ; Copy descriptor to R2-R3
63 52 2A 3A 03BC 762 LOCC #A'*,R2,(R3) ; Asterisk wildcard in string?
09 12 03C0 763 BNEQ 145$ ; Branch if found
63 52 25 3A 03C2 764 LOCC #A'%',R2,(R3) ; Percent wildcard in string?
03 12 03C6 765 BNEQ 145$ ; Branch if found
58 04 CA 03C8 766 BICL2 #4,R8 ; Indicate nonwild terminator string
63 2A 91 03CB 767 145$: CMPB #A'*,(R3) ; Leading asterisk wildcard?
03 13 03CE 768 BEQL 146$ ; Branch if yes
58 08 CA 03D0 769 BICL2 #8,R8 ; Indicate non-asterisk terminator
69 52 04 A1 03D3 770 146$: ADDW3 #4,R2,(R9) ; Set byte count of name and type
04 B9 63 52 28 03D7 771 MOVW R2,(R3),a4(R9) ; Set terminator file name
63 5249442E 8F D0 03DC 772 MOVL #A'.DIR',(R3) ; Set terminator file type
10 BC 01 B0 03E3 773 MOVW #1,a16(AP) ; Set terminator file version
50 58 D0 03E7 774 150$: MOVL R8,R0 ; Set return status
04 03EA 775 RET ; Return
```



```
03EB 777      .SBTTL TERMINATE_SCAN, test for scan termination
03EB 778
03EB 779      :++
03EB 780      :
03EB 781      : Functional Description:
03EB 782      : This routine executes a greater-than match on the name, type, and
03EB 783      : version of a candidate and terminator file specification.
03EB 784      :
03EB 785      : Calling Sequence:
03EB 786      : CALLS/CALLG
03EB 787      :
03EB 788      : Input Parameters:
03EB 789      : 04(AP) = Length of candidate name and type string.
03EB 790      : 08(AP) = Address of candidate name and type string.
03EB 791      : 12(AP) = Candidate version number.
03EB 792      : 16(AP) = Length of terminator name and type string.
03EB 793      : 20(AP) = Address of terminator name and type string.
03EB 794      : 24(AP) = Terminator version number.
03EB 795      :
03EB 796      : Implicit Inputs:
03EB 797      : none
03EB 798      :
03EB 799      : Output Parameters:
03EB 800      : none
03EB 801      :
03EB 802      : Implicit Outputs:
03EB 803      : none
03EB 804      :
03EB 805      : Routines Called:
03EB 806      : none
03EB 807      :
03EB 808      : Routine Value:
03EB 809      : True if the candidate file specification is known to follow the
03EB 810      : terminator file specification in an ODS-2 directory.
03EB 811      :
03EB 812      : Signals:
03EB 813      : none
03EB 814      :
03EB 815      : Side Effects:
03EB 816      : none
03EB 817      :
03EB 818      :--
003C 03EB 819
03EB 820      .ENTRY TERMINATE_SCAN,*M<R2,R3,R4,R5>
03ED 821      :
03ED 822      : Initialize for the loop.
03ED 823      :
03ED 824      CLRL    R0                ; Assume failure return
52   04 AC 7D 03EF 825      MOVQ    4(AP),R2          ; Get candidate string descriptor
54   10 AC 7D 03F3 826      MOVQ    16(AP),R4         ; Get terminator string descriptor
03F7 827      :
03F7 828      : Main scanning loop.
03F7 829      :
03F7 830 10$:  DECL    R4                ; Terminator exhausted?
51   1C 19 03F9 831      BLSS     40$,              ; Branch if yes
2A   51 9A 03FB 832      MOVZBL   (R5)+,R1          ; Get next character in terminator
03FE 833      CMPB     R1,#'A'*,              ; Terminator specifies wild string?
```

```
25 13 13 0401 834 BEQL 30$ ; Branch if yes
    51 91 0403 835 CMPB R1, #^A'^ ; Terminator specifies wild character?
    OE 13 0406 836 BEQL 30$ ; Branch if yes
    52 D7 0408 837 DECL R2 ; Candidate exhausted?
    OA 19 040A 838 BLSS 30$ ; Branch if yes
83 51 91 040C 839 CMPB R1, (R3)+ ; Compare terminator to candidate
    E6 13 040F 840 BEQL 10$ ; Branch if terminator equals candidate
    03 1A 0411 841 BGTRU 30$ ; Branch if terminator greater
50 01 D0 0413 842 20$: MOVL #1, R0 ; Set success return
    04 0416 843 30$: RET ; Return
    0417 844 ;
    0417 845 ; Here when terminator is exhausted.
    0417 846 ;
    52 D5 0417 847 40$: TSTL R2 ; Candidate exhausted?
    F8 12 0419 848 BNEQ 20$ ; Branch if no
    041B 849 ;
    041B 850 ; The match succeeded. Now the version numbers must be matched.
    041B 851 ;
18 AC OC AC D1 041B 852 CMPL 12(AP), 24(AP) ; Compare candidate to terminator
    F1 19 0420 853 BLSS 20$ ; Branch if candidate smaller
    04 0422 854 RET ; Return failure
```



```
0423 856 .SBTTL PARSE_DIRECTORY, parse directory into components
0423 857
0423 858 :++
0423 859
0423 860 : Functional Description:
0423 861 : This routine parses the directory portion of a file specification.
0423 862
0423 863 : Calling Sequence:
0423 864 : JSB
0423 865
0423 866 : Input Parameters:
0423 867 : R2 = Length of directory string.
0423 868 : R3 = Address of directory string.
0423 869 : R4 = Pointer to result area.
0423 870
0423 871 : Implicit Inputs:
0423 872 : none
0423 873
0423 874 : Output Parameters:
0423 875 : none
0423 876
0423 877 : Implicit Outputs:
0423 878 : Result area contains a count of descriptors followed by one descriptor
0423 879 : for each component of the directory specification. An ellipsis is
0423 880 : represented by a one-byte string '..'. If the directory is in UIC
0423 881 : format, bit 16 of the descriptor count longword is set.
0423 882
0423 883 : Routines Called:
0423 884 : none
0423 885
0423 886 : Routine Value:
0423 887 : none
0423 888
0423 889 : Signals:
0423 890 : none
0423 891
0423 892 : Side Effects:
0423 893 : R0-R5 destroyed.
0423 894
0423 895 :--
0423 896
0423 897 PARSE_DIRECTORY:
0423 898 : MOVAL (R4)+,R5 ; Keep pointer to result area and bump
0423 899 : CLRL (R5) ; Clear component count
0423 900
0423 901 : Main scanning loop.
0423 902
0423 903 10$: LOCC #^A'..',R2,(R3) ; Scan for delimiter
0423 904 : BEQL 20$ ; Branch if none found
0423 905 : SUBL3 R0,R2,(R4)+ ; Set length of this component
0423 906 : MOVL R3,(R4)+ ; Set address of this component
0423 907 : INCL (R5) ; Count this component
0423 908 : MOVAB -(R0),R2 ; Prune this component from string
0423 909 : MOVAB 1(R1),R3
0423 910 : CMPL R2,#2 ; At least 2 characters left?
0423 911 : BLSS 10$ ; Branch if no
0423 912 : CMPW (R3),#^A'..' ; Ellipsis present?
```

55	84	DE	0423	898	MOVAL	(R4)+,R5		: Keep pointer to result area and bump
	65	D4	0426	899	CLRL	(R5)		: Clear component count
			0428	900				
			0428	901				
			0428	902				
63	52	2E	3A	0428	903	10\$: LOCC	#^A'..',R2,(R3)	: Scan for delimiter
		29	13	042C	904	: BEQL	20\$	: Branch if none found
84	52	50	C3	042E	905	: SUBL3	R0,R2,(R4)+	: Set length of this component
	84	53	D0	0432	906	: MOVL	R3,(R4)+	: Set address of this component
		65	D6	0435	907	: INCL	(R5)	: Count this component
	52	70	9E	0437	908	: MOVAB	-(R0),R2	: Prune this component from string
53	01	A1	9E	043A	909	: MOVAB	1(R1),R3	
	02	52	D1	043E	910	: CMPL	R2,#2	: At least 2 characters left?
		E5	19	0441	911	: BLSS	10\$	: Branch if no
2E2E	8F	63	B1	0443	912	: CMPW	(R3),#^A'..'	: Ellipsis present?

```
52 DE 12 0448 913 BNEQ 10$ : Branch if no
84 02 C2 044A 914 SUBL2 #2,R2 : Adjust count
84 01 D0 044D 915 MOVL #1,(R4)+ : Set length of this component
84 83 3E 0450 916 MOVAW (R3)+,(R4)+ : Set address of '.'
65 D6 0453 917 INCL (R5) : Count this component
D1 11 0455 918 BRB 10$ : Branch to get next component
0457 919 :
0457 920 : Here when no '.' found in string.
0457 921 :
52 D5 0457 922 20$: TSTL R2 : At end of string?
05 13 0459 923 BEQL 30$ : Branch if no
84 52 7D 045B 924 MOVQ R2,(R4)+ : Set descriptor for last component
65 D6 045E 925 INCL (R5) : Count last component
01 65 D1 0460 926 30$: CML (R5),#1 : One component?
5D 12 0463 927 BNEQ 70$ : Branch if no
52 04 A5 7D 0465 928 MOVQ 4(R5),R2 : Get descriptor for first component
63 52 2C 3A 0469 929 LOCC #^A',R2,(R3) : In UIC format?
53 13 046D 930 BEQL 70$ : Branch if no
046F 931 :
046F 932 : Special processing for UIC-format directory.
046F 933 :
02 A5 01 88 046F 934 BISB2 #1,2(R5) : Set UIC format bit
54 0C A5 9E 0473 935 MOVAB 12(R5),R4 : Point to UIC storage
04 A5 06 D0 0477 936 MOVL #6,4(R5) : Reset component descriptor
08 A5 54 D0 047B 937 MOVL R4,8(R5) :
84 30303030 8F D0 047F 938 MOVL #^A'0000',(R4)+ : Initialize to 000000
84 3030 8F B0 0486 939 MOVW #^A'00',(R4)+ :
53 52 C0 048B 940 ADDL2 R2,R3 : Point past member number
52 50 C2 048E 941 SUBL2 R0,R2 : Get length of group number
50 D7 0491 942 DECL R0 : Get length of member number
2A FF A3 91 0493 943 CMPB -1(R3),#^A'*' : Is member number '*'?
74 2525 8F B0 0499 944 BNEQ 40$ : Branch if no
74 25 90 049E 945 MOVW #^A'%%',-(R4) : Make it '%%' in output
74 06 11 04A1 946 MOVW #^A'%' ,-(R4) :
74 73 90 04A3 947 BRB 50$ : Done
FA 50 F5 04A6 948 40$: MOVW -(R3),-(R4) : Copy one digit
54 OF A5 9E 04A9 949 SOBGR R0,40$ : Loop until done
2A FF A1 91 04AD 950 50$: MOVAB 15(R5),R4 : Point past group number
74 2525 8F B0 04B3 951 CMPB -1(R1),#^A'*' : Is group number '*'?
74 25 90 04B8 952 BNEQ 60$ : Branch if no
74 71 90 04BC 953 MOVW #^A'%%',-(R4) : Make it '%%' in output
FA 52 F5 04BF 954 MOVW #^A'%' ,-(R4) :
05 04BB 955 RSB : Done
74 71 90 04BC 956 60$: MOVW -(R1),-(R4) : Copy one digit
FA 52 F5 04BF 957 SOBGR R2,60$ : Loop till done
05 04C2 958 70$: RSB : Done
04C3 959
04C3 960 .END
```



MATCH  
Symbol table

File specification matching

H 6

15-SEP-1984 23:37:13 VAX/VMS Macro V04-00  
4-SEP-1984 22:59:09 [BACKUP.SRC]MATCH.MAR;1

Page 22  
(9)

FMGSMATCH NAME	*****	X	02
INIT_NAMEBLOCK	00000000	RG	02
INIT_SEL_INFO	00000145	RG	02
LIBSCVT_DTB	*****	X	02
MATCH	00000197	RG	02
MATCH_DIRECTORY	0000027A	RG	02
MATCH_FILENAME	00000209	RG	02
NAMSB_DEV	= 00000039		
NAMSB_DIR	= 0000003A		
NAMSB_ESL	= 0000000B		
NAMSB_NAME	= 0000003B		
NAMSB_NODE	= 00000038		
NAMSB_RSL	= 00000003		
NAMSB_TYPE	= 0000003C		
NAMSB_VER	= 0000003D		
NAMSL_DEV	= 00000044		
NAMSL_DIR	= 00000048		
NAMSL_ESA	= 0000000C		
NAMSL_FNB	= 00000034		
NAMSL_NAME	= 0000004C		
NAMSL_NODE	= 00000040		
NAMSL_RSA	= 00000004		
NAMSL_TYPE	= 00000050		
NAMSL_VER	= 00000054		
NAMSV_GRP_MBR	= 00000013		
NAMSV_WILD_GRP	= 00000018		
NAMSV_WILD_MBR	= 00000019		
NAMSV_WILD_UFD	= 00000018		
PARSE_DIRECTORY	00000423	R	02
TERMINATE_SCAN	000003EB	RG	02
WILD_DIRECTORY	00000077	RG	02

+-----+  
! Psect synopsis !  
+-----+

PSECT name

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 ( 0.)	00 ( 0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS\$	00000000 ( 0.)	01 ( 1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
CODE	000004C3 ( 1219.)	02 ( 2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC	BYTE				

+-----+  
! Performance indicators !  
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	20	00:00:00.14	00:00:00.88
Command processing	107	00:00:00.66	00:00:02.17
Pass 1	130	00:00:03.75	00:00:07.51
Symbol table sort	0	00:00:00.23	00:00:00.37
Pass 2	163	00:00:01.91	00:00:04.95
Symbol table output	5	00:00:00.03	00:00:00.03
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	426	00:00:06.74	00:00:15.95

The working set limit was 1200 pages.  
24889 bytes (49 pages) of virtual memory were used to buffer the intermediate code.  
There were 20 pages of symbol table space allocated to hold 168 non-local and 56 local symbols.  
960 source lines were read in Pass 1, producing 34 object records in Pass 2.  
8 pages of virtual memory were used to define 7 macros.

↑-----↑  
! Macro library statistics !  
↓-----↓

Macro library name

Macros defined

-----  
\_S255SDUA28:[SYSLIB]STARLET.MLB;2

-----  
4

217 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:MATCH/OBJ=OBJ\$:MATCH MSRC\$:MATCH/UPDATE=(ENH\$:MATCH)



0012 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

